CLAIMS:

1. A method for improving the performance of a cache comprising the steps of:

receiving a request of an address of data;

determining if said requested data is located in said cache;

determining if said requested address is associated with contents of a first tag if said requested data is not located in said cache; and

replacing a data element in said cache using said contents of said first tag if said requested address is associated with said contents of said first tag and if a bit associated with said first tag is in a first state, wherein said bit associated with said first tag is in said first state if said contents of said first tag were previously requested and not located in said cache.

2. The method as recited in claim 1 further comprising the step of:

setting said bit associated with said first tag to a second state if said requested data is located in said cache.

3. The method as recited in claim 1 further comprising the step of:

replacing said contents of said first tag with contents of a second tag if said requested address is not associated with said contents of said first tag.

4. The method as recited in claim 3 further comprising the step of:

resetting said bit associated with said first tag to said first state if said requested address is not associated with said contents of said first tag.

5. The method as recited in claim 1 further comprising the step of:

resetting said bit associated with said first tag to said first state if said requested address is associated with said contents of said first tag and if said bit associated with said first tag is in a second state.

6. The method as recited in claim 1 further comprising the step of:

loading contents of a second tag into said contents of said first tag if said requested address is associated with said contents of said first tag and said bit associated with said first tag is in a second state.

7. The method as recited in claim 6 further comprising the step of:

resetting said bit associated with said first tag to said first state if said requested address is associated with said contents of said first tag.

8. The method as recited in claim 1 further comprising the step of:

loading said contents of said first tag into contents of a second tag if said requested address is associated with said contents of said first tag and said bit associated with said first tag is in a second state.

9. The method as recited in claim 1 further comprising the step of:

determining if a counter equals a maximum value if said requested address is not associated with said contents of said first tag, wherein said counter tracks a number of times said contents of said first tag is replaced without being reset.

10. The method as recited in claim 9 further comprising the step of:

updating said counter if said counter does not equal said maximum value.

11. A system, comprising:

a processor, wherein said processor has a cache memory associated with it;

a system memory for storing data of said processor;

a bus system coupling said processor to said system memory;

wherein said cache memory comprises:

a first tag storing an indication of an address previously requested by said processor not stored in said cache memory;

logic for receiving a request of an address of data from said processor;

logic for determining if said requested data is located in said cache memory;

logic for determining if said requested address is associated with contents of said first tag if said requested data is not located in said cache; and

logic for replacing a data element in said cache using said contents of said first tag if said requested address is associated with said contents of said first tag and if a bit associated with said first tag is in a first state, wherein said bit associated with said first tag is in said first state if said contents of said first tag were previously requested and not located in said cache.

12. The system as recited in claim 11, wherein said cache memory further comprises:

logic for setting said bit associated with said first tag to a second state if said requested data is located in said cache.

13. The system as recited in claim 11, wherein said cache memory further comprises:

logic for replacing said contents of said first tag with contents of a second tag if said requested address is not associated with said contents of said first tag, wherein said second tag is associated with a data element in said cache memory.

14. The system as recited in claim 13, wherein said cache memory further comprises:

logic for resetting said bit associated with said first tag to said first state if said requested address is not associated with said contents of said first tag.

15. The system as recited in claim 11, wherein said cache memory further comprises:

logic for resetting said bit associated with said first tag to said first state if said requested address is associated with said contents of said first tag and if said bit associated with said first tag is in a second state.

16. The system as recited in claim 11, wherein said cache memory further comprises:

logic for loading contents of a second tag into said contents of said first tag if said requested address is associated with said contents of said first tag and said bit associated with said first tag is in a second state, wherein said second tag is associated with a data element in said cache memory.

17. The system as recited in claim 16, wherein said cache memory further comprises:

logic for resetting said bit associated with said first tag to said first state if said requested address is associated with said contents of said first tag.

18. The system as recited in claim 11, wherein said cache memory further comprises:

logic for loading said contents of said first tag into contents of a second tag if said requested address is associated with said contents of said first tag and said bit associated with said first tag is in a second state, wherein said second tag is associated with a data element in said cache memory.

19. The system as recited in claim 11, wherein said cache memory further comprises:

logic for determining if a counter equals a maximum value if said requested address is not associated with said contents of said first tag, wherein said counter tracks a number of times said contents of said first tag is replaced without being reset.

20. The system as recited in claim 19, wherein said cache memory further comprises:

logic for updating said counter if said counter does not equal said maximum value.